UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

| APPLICATION NO. | FILING DATE | FIRST NAMED INVENTOR | ATTORNEY DOCKET NO. | CONFIRMATION NO. |
|---|---|---|---|---|
| 09/806,743 | 04/02/2001 | Timothy Edward Miller | 8224 | 4708 |

26890 7590 04/08/2005

JAMES M. STOVER
NCR CORPORATION
1700 SOUTH PATTERSON BLVD, WHQ4
DAYTON, OH 45479

| EXAMINER | |
|---|---|
| HOLMES, MICHAEL B | |
| **ART UNIT** | **PAPER NUMBER** |
| 2121 | |

DATE MAILED: 04/08/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

| Office Action Summary | Application No. | Applicant(s) |
|---|---|---|
| | 09/806,743 | MILLER ET AL. |
| | Examiner | Art Unit |
| | Michael B. Holmes | 2121 |

*-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --*

**Period for Reply**

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

**Status**

1)☒ Responsive to communication(s) filed on *12 January 2005*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

**Disposition of Claims**

4)☒ Claim(s) *1-83* is/are pending in the application.

4a) Of the above claim(s) *1-23* is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *24-83* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

**Application Papers**

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *02 April 2001* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

**Priority under 35 U.S.C. § 119**

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

**Attachment(s)**

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☒ Other: *Detailed Office Action*.

**UNITED STATES PATENT AND TRADEMARK OFFICE**
P.O. Box 1450, Alexandria, Virginia 22313-1450 – WWW.USPTO.GOV

# Examiner's Detailed Office Action

1. This office action is responsive to communication received on January 12, 2005.

2. Claims 1-23 have been canceled.

3. Claims 24-83 have been added and examined.

## Claim Rejections - 35 USC § 102

4. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in a patent granted on an application for patent by another filed in the United States before the invention thereof by the applicant for patent, or on an international application by another who has fulfilled the requirements of paragraphs (1), (2), and (4) of section 371(c) of this title before the invention thereof by the applicant for patent.

5. The changes made to 35 U.S.C. 102(e) by the American Inventors Protection Act of 1999 (AIPA) and the Intellectual Property and High Technology Technical Amendments Act of 2002 do not apply when the reference is a U.S. patent resulting directly or indirectly from an international application filed before November 29, 2000. Therefore, the prior art date of the reference is determined under 35 U.S.C. 102(e) prior to the amendment by the AIPA (pre-AIPA 35 U.S.C. 102(e)).

6. Claims 25-34, 36-40, 42, 43, 46-55, 57-61, 63-74, 76-80, 82 & 83 are rejected under 35 U.S.C. 102(e) as being anticipated by *Iyer et al.* (USPN 5,899,992).

Regarding claim 24:

*Iyer et al.* teaches,

A computer-implemented system for performing data mining applications, comprising:

**(a)** a computer having one or more data storage devices connected thereto, wherein a relational database is stored on one or more of the data storage devices **[(FIG. 1;** item **104** *random access memory (RAM)*) **& (col. 3, line 9-29 "*The RDBMS software 108*** *receives commands from users for performing various search and retrieval functions, termed queries, against one or more databases 112 stored in the data storage devices 106. In the preferred embodiment, these queries conform to the Structured Query Language (SQL) standard, although other types of queries could also be used without departing from the scope of the invention. The queries invoke functions performed by the* ***RDBMS software 108****, such as definition, access control, interpretation, compilation, database retrieval, and update of user and system data.* ***Generally, the RDBMS software 108****, the SQL queries, and the instructions derived therefrom, are all tangibly embodied in or readable from a computer-readable medium, e.g. one or more of the data storage devices 106 and/or data communications devices coupled to the computer.* ***Moreover, the RDBMS software 108****, the SQL queries, and the instructions derived therefrom, are all comprised of instructions which, when read and executed by the computer 100, causes the computer 100 to perform the steps necessary to implement and/or use the present invention.*"**)]**;

**(b)** a relational database management system, executed by the computer, for accessing the relational database stored on the data storage devices **[(col. 2, line 54 to col. 3, line 9**

"*FIG. 1 is a block diagram illustrating an exemplary hardware environment used to implement the preferred embodiment of the invention. In the exemplary environment, a computer 100 is comprised of one or more processors 102, random access memory (RAM) 104, and assorted peripheral devices. The peripheral devices usually include one or more fixed and/or removable data storage devices 106, such as a hard disk, floppy disk, CD-ROM, tape, etc. Those skilled in the art will recognize that any combination of the above components, or any number of different components, peripherals, and other devices, may be used with the computer 100. The present invention is typically implemented using* ***relational database management system (RDBMS) software 108****, such as the DB2 product sold by IBM Corporation, although it may be imple mented with any database management system (DBMS) software.* ***The RDBMS software 108*** *executes under the control of an operating system 110, such MVS, AIX, OS/2, WINDOWS NT, WINDOWS, UNIX, etc. Those skilled in the art will recognize that any combination of the software, or any number of different software, may be used to implement the present invention.*"**)]**; and **(c)** an analytic application programming interface (API) that generates a set of scalable data mining functions including queries for execution by the relational database management system, executed by the computer, for performing data mining operations directly within the database management system. **[(col. 3, line 50 to col. 4, line 26 "*The scalable set-oriented classifier 114*** *of the present invention* ***resorts to proven scalable database technology*** *to provide a generic* ***solution to the classification problem of scalability. The present invention provides a scalable model for classifying rows of a table within a classification tree.*** *The* ***scalable set-oriented classifier 114 is called the Scalable Supervised Learning Irregardless of Memory (SLIM) Classifier 114****. Not only is the SLIM classifier 114 scalable in regions where recently published*

*classifiers are not, but by virtue of building on well known set-oriented database management system (DBMS) primitives, the SLIM classifier 114 instantly exploits several decades of database research and development. The present invention rephrases classification, a data mining method, into analysis of data in a star schema, formalizing further the interrelationship between data mining and data warehousing. A description of a prototype built using* ***IBM's DB2 product as the RDBMS 108****, and experimental results for the prototype are discussed below. Generally, the experimental results indicate that the DB2-based SLIM classifier 114 has desirable properties associating it with linear scalability. The SLIM classifier 114 is built based on a set-oriented access to data paradigm. The SLIM classifier 114 uses Structured Query Language (SQL), offered by most commercial RDBMS 108 vendors, as the basis for the method. The SLIM classifier 114 is based on* ***well known database methodologies and lets the RDBMS 108*** *automatically handle scalability. As a result, the SLIM classifier 114 will scale as long as the database scales. The SLIM classifier 114 leverages the Structured Query Language (SQL)* ***Application Programming Interface (API) of the RDBMS 108, which exploits the benefits of many years research and development pertaining to: (1) scalability*** *(2) memory hierarchy (3)* ***parallelism*** *([18]) (4) optimization of the executions([16]) (5) platform independence* ***(6) client server API ([17]).***")]

Regarding claims 25, 46 & 65:

*Iyer et al.* teaches,

wherein the computer comprises a parallel processing computer comprised of a plurality of nodes, and each node executes one or more threads of the relational database management

system to provide parallelism in the data mining operations. [**Abstract** ("*A method, apparatus, and article of manufacture for a computer implemented scaleable set oriented classifier. The scalable set-oriented classifier stores set-oriented data as* ***a table in a relational database****. The table is comprised of rows having attributes. The scalable set-oriented classifier classifies the rows by building* ***a classification tree****. The scalable set-oriented classifier determines a gini index value for each split value of each attribute for each* ***node*** *that can be partitioned in the* ***classification tree****. The scalable set-oriented classifier selects an attribute and a split value for each* ***node*** *that can be partitioned based on the determined gini index value corresponding to the split value. Then, the scalable set-oriented classifier grows the classification tree by another level based on the selected attribute and split value for each* ***node****. The scalable set-oriented classifier repeats this process until each row of the table has been classified in the* ***classification tree****.*") & (**col. 4, line 12-22** "*The SLIM classifier 114 is based on* ***well known database methodologies and lets the RDBMS 108*** *automatically handle scalability. As a result, the SLIM classifier 114 will scale as long as the database scales. The SLIM classifier 114 leverages the Structured Query Language (SQL)* ***Application Programming Interface (API) of the RDBMS 108, which exploits the benefits of many years research and development pertaining to****: (1)* ***scalability*** *(2) memory hierarchy (3)* ***parallelism*** *([18]) (4) optimization of the executions([16]) (5) platform independence* ***(6) client server API ([17])****.*")]

Regarding claims 26, 47 & 66:

*Iyer et al.* teaches,

wherein the scalable data mining functions process data collections stored in the relational database and produce results that are stored in the relational database. **[Abstract (**"*A method, apparatus, and article of manufacture for a computer implemented* ***scaleable*** *set-oriented classifier. The* ***scalable*** *set-oriented classifier* ***stores*** *set-oriented data as* ***a table in a relational database****. The table is comprised of rows having attributes. The* ***scalable*** *set-oriented classifier classifies the rows by building* ***a classification tree****. The* ***scalable*** *set-oriented classifier determines a gini index value for each split value of each attribute for each* ***node*** *that can be partitioned in the* ***classification tree****. The* ***scalable*** *set-oriented classifier selects an attribute and a split value for each* ***node*** *that can be partitioned based on the determined gini index value corresponding to the split value. Then, the* ***scalable*** *set-oriented classifier grows the classification tree by another level based on the selected attribute and split value for each* ***node****. The* ***scalable*** *set-oriented classifier repeats this process until each row of the table has been classified in the* ***classification tree.***"**)]**

Regarding claims 27, 48 & 67:

*Iyer et al.* teaches,

wherein the scalable data mining functions are created by parameterizing and instantiating the analytic API. **[(col. 2, line 15-27** "*The scalable set-oriented classifier classifies the rows by building a classification tree.* ***The scalable set-oriented classifier determines a gini index value for each split value of each attribute for each node that can be partitioned in the classification***

***tree****. The scalable set-oriented classifier selects an attribute and a split value for each node that can be partitioned based on the determined* ***gini index value*** *corresponding to the split value. Then, the scalable set-oriented classifier grows the classification tree by another level based on the selected attribute and split value for each node. The scalable set-oriented classifier repeats this process until each row of the table has been classified in the classification tree.*") **& (col. 4, line 12-22** "*The SLIM classifier 114 is based on* ***well known database methodologies and lets the RDBMS 108*** *automatically handle scalability. As a result, the SLIM classifier 114 will scale as long as the database scales. The SLIM classifier 114 leverages the Structured Query Language (SQL)* ***Application Programming Interface (API) of the RDBMS 108, which exploits the benefits of many years research and development pertaining to****: (1)* ***scalability*** *(2) memory hierarchy (3)* ***parallelism*** *([18]) (4) optimization of the executions([16]) (5) platform independence* ***(6) client server API ([17])***."**)]**

Regarding claims 28, 49 & 68:

*Iyer et al.* teaches,

wherein the scalable data mining functions are dynamically generated queries comprised of combined phrases with substituting values therein based on parameters supplied to the analytic API. **[(col. 4, line 04-22** "*The SLIM classifier 114 is built based on a set-oriented access to data paradigm. The SLIM classifier 114 uses* ***Structured Query Language (SQL)****, offered by most commercial RDBMS 108 vendors, as the basis for the method. The SLIM classifier 114 is based on well known database methodologies and lets the RDBMS 108 automatically handle scalability. As a result, the SLIM classifier 114 will scale as long as the database scales. The*

*SLIM classifier 114 is based on* ***well known database methodologies and lets the RDBMS 108*** *automatically handle scalability. As a result, the SLIM classifier 114 will scale as long as the database scales.* ***The SLIM classifier 114 leverages the Structured Query Language (SQL) Application Programming Interface (API) of the RDBMS 108, which exploits the benefits of many years research and development pertaining to: (1) scalability*** *(2) memory hierarchy (3)* ***parallelism*** *([18]) (4)* ***optimization of the executions*** *([16]) (5) platform independence* ***(6) client server API*** ***([17])***."")]

Regarding claims 29, 50 & 69:

*Iyer et al.* teaches,

wherein the scalable data mining functions are selected from a group of functions comprising Data Description functions, Data Derivation functions, Data Reduction functions, Data Reorganization functions, **Data Sampling functions, and Data Partitioning functions.**

**[Abstract** (**"***A method, apparatus, and article of manufacture for a computer implemented* ***scaleable*** *set-oriented classifier. The* ***scalable*** *set-oriented classifier* ***stores*** *set-oriented data as* ***a table in a relational database****. The table is comprised of rows having attributes. The* ***scalable*** *set-oriented classifier classifies the rows by building* ***a classification tree. The scalable set-oriented classifier determines a gini index value for each split value of each attribute for each node that can be partitioned in the classification tree****. The* ***scalable*** *set-oriented classifier selects an attribute and a split value for each* ***node*** *that can be* ***partitioned based on the determined gini index value corresponding to the split value****. Then, the* ***scalable*** *set-oriented classifier grows the classification tree by another level based on the selected attribute and split*

*value for each* ***node****. The* ***scalable*** *set-oriented classifier repeats this process until each row of the table has been classified in the* ***classification tree.***"**)]**

Regarding claims 30, 51 & 70:

*Iyer et al.* teaches,

wherein the Data Description functions comprise descriptive statistical functions. **[Abstract ("***A method, apparatus, and article of manufacture for a computer implemented scaleable set-oriented classifier. The scalable set-oriented classifier stores set-oriented data as a table in a relational database. The table is comprised of rows having attributes. The* ***scalable*** *set-oriented classifier classifies the rows by building a classification tree. The scalable set-oriented classifier determines a* ***gini index value*** *(Examiner interprets the gini index as the statistical function) for each split value of each attribute for each node that can be partitioned in the classification tree. The scalable set-oriented classifier selects an attribute and a split value for each* ***node*** *that can be partitioned based on the determined* ***gini index value*** *corresponding to the split value. Then, the scalable set-oriented classifier grows the classification tree by another level based on the selected attribute and split value for each node. The scalable set-oriented classifier repeats this process until each row of the table has been classified in the classification tree.*"**)]**

Regarding claims 31, 52 & 71:

*Iyer et al.* teaches,

wherein the Data Description functions are selected from a group comprising:

(1) descriptive statistics for one or more numeric columns, wherein the statistics are selected

from a group comprising count, minimum, maximum, mean, standard deviation, standard mean error, variance, coefficient of variance, skewness, kurtosis, uncorrected sum of squares, corrected sum of squares, and quantiles,

(2) a count of values for a column, **[(col. 9, line 34-39** "*Similarly, the DOWN table could be generated by just changing the <=to> in the ON clause. Also, the SLIM classifier 114 can obtain the DOWN table by using the information in the leaf nodes and* ***the count column*** *in the UP table without doing join on DIMI again.*"**)]**

(3) a calculated modality for a column,

(4) one or more bin numeric columns of counts with overlay and statistics options,

(5) one or more automatically sub-burned numeric columns giving additional counts and isolated frequently occurring individual values

(6) a computed frequency of one or more column values, (7) a computed frequency of values for pairs of columns in a column list,

(8) a Pearson Product-Moment Correlation matrix,

(9) a Covariance matrix,

(10) a sum of squares and cross-products matrix, and

(11) a count of overlapping column values in one or more combinations of tables.

Regarding claims 32, 53 & 72:

*Iyer et al.* teaches,

wherein the Data Derivation functions provide column derivations or transformations. [**FIG. 4**; (col. 10, line 07-39 "*In step 450, the SLIM classifier 114* ***calculates the gini index for each***

***possible split value for attribute i.*** *Now a view GINI.sub.__VALUE that contains all gini index values at each possible split value is generated. Taking the liberty with SQL syntax, the following query is written: ... Note the* ***transformation*** *for the table name DIM.sub.i to* ***column value i*** *and* ***column name attr.sub.i.*** *... The MIN.sub.__GINI table contains the best split value and the corresponding* ***gini index value*** *for each leaf node of the classification tree 200 with respect to attribute i.*")]

Regarding claims 33, 54 & 73:

*Iyer et al.* teaches,

wherein the Data Description functions are selected from a group comprising:

(1) a derived binned numeric column wherein a new column is bin number,

(2) a n-valued categorical column dummy-coded into "n" 0/1 values,

(3) a n-valued categorical column recoded into n or less new values,

(4) one or more numeric columns scaled via range transformation,

(5) one or more columns scaled to a z-score that is a number of standard deviations from a mean,

(6) one or more numeric columns scaled via a sigmoidal transformation function,

(7) one or more numeric columns scaled via a base 10 logarithm function,

(8) one or more numeric columns scaled via a natural logarithm function,

(9) one or more numeric columns scaled via an exponential function,

(10) one or more numeric columns raised to a specified power,

(11) one or more numeric columns derived via user defined transformation function,

(12) one or more new columns derived by ranking one or more columns or expressions based on order, **[(col. 9, line 14-39 "*The new operator forms multiple groupings concurrently, and may allow further optimization*.** *For each non-STOP leaf node in the tree, possible split values for attribute i are all distinct values of attr.sub.i among the examples which belong to this leaf node. For each possible split value, the SLIM classifier 114 needs to get the class distribution for the two parts partitioned by this value to compute the corresponding gini index. In step 430, the SLIM classifier 114 collects such distribution information n two tables, UP and DOWN. ... Similarly, the DOWN table could be generated by just changing the <=to> in the ON clause. Also, the SLIM classifier 114 can obtain the DOWN table by using the information in the leaf nodes and* ***the count column in the UP table*** *without doing join on DIMI again.*"**)]**

(13) one or more new columns derived with quantile 0 to n-1 based on order and n,

(14) a cumulative sum of a value expression based on a sort expression,

(15) a moving average of a value expression based on a width and order,

(16) a moving sum of a value expression based on a width and order,

(17) a moving difference of a value expression based on a width and order,

(18) a moving linear regression value derived from an expression, width, and order,

(19) a multiple account/product ownership bitmap,

(20) a product ownership bitmap over multiple time periods,

(21) one or more counts, amount, percentage means and intensities derived from a transaction summary,

(22) one or more variabilities derived from transaction summary data,

(23) one or more derived trigonometric values and their inverses, including sin, arcsin,

cos, arccos, csc, arccsc, sec, arcsec, tan, arctan, cot, and arccot, and

(24) one or more derived hyperbolic values and their inverses, including sinh, arcsinh, cosh, arccosh, csch, arccsch, sech, arcsech, tanh, arctanh, coth, and arccoth.

Regarding claims 34, 55 & 74:

*Iyer et al.* teaches,

wherein the Data Reduction functions provide matrix building operations to reduce the amount of data required for analytic algorithms. **[(col. 10, line 64 to col. 11, line 10 "***For a categorical attribute i, the SLIM classifier 114 forms DIM.sub.1 in the same way as for a numerical attribute. DIM.sub.i contains all the information the SLIM classifier 114 needs to compute the gini index for any subset splitting. In fact, It is an analog of the* ***count matrix*** *in Shafer, but formed with set-oriented operators.* ***A possible split is any subset of the set that contains all the distinct attribute values****. If the cardinality of attribute i is m, the SLIM classifier 114 needs to evaluate the splits for all the ...subsets. Those subsets and their related counts can be generated in a recursive way. ...follows: ...***")]**

Regarding claims 36, 57 & 76:

*Iyer et al.* teaches,

wherein the Data Reorganization functions provide an ability to reorganize data by joining or de-normalizing pre-processed results into a wide analytic data set. **[(col. 9, line 39-50 "***In case the* ***outer-join operator*** *is not supported, by performing simple set operations such as EXCEPT and UNION, the SLIM classifier 114 can form a view DIM.sub.i with the same schema as DIM.sub.i*

*first. For each possible split value on attribute i and each possible class label of each node, there is a row in DIM.sub.i that gives the number of rows belonging to this leaf node that have such a value on attribute i and such a class label. Note that DIM.sub.i is a superset of DIM.sub.i and the difference between them are those rows with a count 0. After DIM.sub.i is generated, the SLIM classifier 114 performs a* ***self-join*** *on DIM.sub.i to create the UP table as follow:*")]

[redacted]

Regarding claims 37, 58 & 77:

[redacted]

*Iyer et al.* teaches,

wherein the Data Reorganization functions are selected from a group comprising:

(1) create a de-normalized new table by removing one or more key columns, and

(2) join a plurality of tables or views into a combined result table. **[(col. 9, line 24-38 "*The UP table with the schema*.** *UP(leaf.sub.__ num, attri, class, count) could be generated by performing a* ***self-outer-join*** *on DIM.sub.i using the following SQL query: ...* ***Similarly, the DOWN table could be generated*** *by just changing the <=to> in the ON clause. Also, the SLIM classifier 114 can obtain the DOWN table by using the information in the leaf nodes and the count column in the UP table without doing* ***join*** *on DIMI again.*")]**

Regarding claims 38, 59 & 78:

*Iyer et al.* teaches,

wherein the Data Sampling function provides an ability to construct a new table containing a randomly selected subset of the rows in an existing table or view. **[(col. 9, line 1-23 "***SELECT FROM DETAIL ...The* ***new operator*** *forms multiple groupings concurrently, and may allow*

*further optimization. For each non-STOP leaf node in the tree, possible split values for attribute i are all distinct values of attr.sub.i among the examples which belong to this leaf node. For each possible split value, the SLIM classifier 114 needs to get the class distribution for the two parts partitioned by this value to compute the corresponding gini index. In step 430, the SLIM classifier 114 collects such distribution information in two tables, UP and DOWN.*")]

Regarding claims 39, 60 & 79:

wherein the Data Sample function selects one or more data samples of specified sizes from a table. **[(col. 14, line 42-55** "*Normally, at this point, the SLIM classifier 114* ***selects*** *the best split value based on the split value of an attribute with the lowest corresponding gini index value. Because both attributes achieve the same gini index value in this example,* ***either one can be selected****. The SLIM classifier 114 stores the best split values in each leaf node of the tree( the root node in this phase). According to the best split value found, the SLIM classifier 114 grows the tree and* ***partitions the training set. The partition*** *is reflected as the leaf.sub.__ num changes in the DETAIL table. Also, any new grown node that is pure or sufficiently small is marked and reassigned a special leaf.sub.__ num value STOP so that the SLIM classifier 114 does not need to process it any more.*")]

Regarding claims 40, 61 & 80:

*Iyer et al.* teaches,

wherein the Data Partitioning function provides an ability to construct a new table containing at least one randomly selected subset of the rows in an existing table or view, wherein the subsets

are mutually distinct but all-inclusive subsets of data. **[(col. 5, line 57** to col. 6. line 8 "*First, **the SLIM classifier 114 initializes a DETAIL table, containing a row for each example in the training set**, and the classification tree 200. Then, until each of the nodes is pure or sufficiently small, the SLIM classifier 114 performs the following procedure. First, for each attribute of an example, **a DIM.sub.i table is generated**. Next, a gini index value is determined for each distinct value (i.e., split value) of each attribute in each leaf node that is to be **partitioned**. Then, the split value with the lowest gini index value is selected for each leaf node that is to be **partitioned** for each attribute i. The best split value for each leaf node that is to be **partitioned** in the classification tree 200 is determined by choosing the attribute with a split value that has the lowest corresponding gini index value for that leaf node. After the best split value is determined, the classification tree 200 is grown by another level. Finally, the nodes that are pure or sufficiently small are marked as "STOP" nodes to indicate that they are not to be **partitioned** any further.*")]

Regarding claims 42, 63 & 82:

*Iyer et al.* teaches,

wherein results of the data mining operations are stored in the relational databases. **[FIG. 1; (col. 3, line 32-34** "*One application of the **RDBMS 108** is known as the **Intelligent Miner(IM) data mining application offered by IBM Corporation** and described in IM User's Guide. The IM is a product consisting of inter-operable kernels and an extensive preprocessing library. The current IM kernels are: Associations, Sequential patterns, Similar time sequences, Classifications, Predicting Values ...*")]

Regarding claims 43, 64 & 83:

*Iyer et al.* teaches,

wherein the relational database management system further comprises an analytical logical data model that stores metadata and processing results from the Scalable Data Mining Functions. **[(col. 6, line 50 to col. 7, line 10** "*There is a one-to-one mapping between leaf.sub.__ num values and leaf nodes in the classification tree 200. If such a mapping is stored in the rows of the DETAIL table, it will be very expensive to access the corresponding leaf node for any row when the table is not memory resident. By examining the mapping carefully, it is seen that the cardinality of the leaf.sub.__ num column is the same as the number of leaf nodes in the classification tree, which is not huge at all, regardless of the* ***size of the training set****. Therefore, the mapping is stored indirectly in a* ***leaf node list (LNL)****. A LNL is a static array that is used to relate the leaf.sub.__ num value in the table to the identification number assigned to the corresponding node in the classification tree 200. By using a labeling technique, the SLIM classifier 114 insures that at each tree growing stage, the nodes always have the identification numbers 0 through N-1, where N is the number of nodes in the tree. LNL[i] is a pointer to the node with identification number i. Now, for any row in the table, the SLIM classifier 114 can get the leaf node it belongs to from its leaf.sub.__ num value and LNL at anytime, and, hence, get the information in the node (e.g. split test, number of examples belonging in this node, and the class distribution of examples belonging in this node). To insure the performance of the SLIM classifier 114, LNL is the only data structure that needs to be memory resident. The* ***size of LNL*** *is equal to the number of nodes in the tree, which is not large at all and which can certainly be stored in memory all the time.* **")]**

# Claim Rejections - 35 USC § 103

9. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

10. Claims 35, 56 & 75 are rejected under 35 U.S.C. 103(a) as being unpatentable over

*Iyer et al.* (USPN 5,899,992) in view of

*SAS Institute Inc.*, SAS OnlineDoc, *Version 8*, Cary, NC: SAS Institute Inc., (09/1999).

The *Iyer et al.* reference has been discussed above and does not explicitly teach the limitation of claims 35, 56 & 75. However, *SAS Institute Inc.* teaches the limitation of claims 35, 56 & 75.

**Regarding Claim 35:**

Regarding claims 35, 56 & 75:

*SAS Institute Inc.* describes, wherein the Data Reduction functions comprise:

(1) build one or more data reduction matrices from a group comprising: (i) **a Pearson-Product Moment Correlations matrix [Figure 40.13 (page1-1)]** It would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matters pertains, to employ a Pearson-Product Moment Correlations matrix or Covariance matrix, because as stated correlation measures the strength of the linear relationship between two variables, moreover, a correlation of 0 means that there is no linear association between two variables, and a correlation of 1 (-1) means that there is an exact positive (negative) linear association between the two variables; (ii) a Covariances matrix; and (iii) a Sum of Squares and Cross Products (SSCP) matrix, (2) export a resultant matrix, and (3) restart a matrix operation.

## Claim Rejections - 35 USC § 103

11. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

12. Claims 41, 62 & 81 are rejected under 35 U.S.C. 103(a) as being unpatentable over *Iyer et al.* (USPN 5,899,992) in view of

SPRINT: A Scalable Parrallel Classifier for Data Mining, *John Shafer, Rakesh Agrawal, Manish Mehta*, Proceeding of the 22nd VLDB Conference Mumbai (Bombay), India, 1996.

The *Iyer et al.* reference has been discussed above and does not explicitly teach the limitation of claims 41, 62 & 81. However, *Shafer et al.* teaches the limitation of claims 41, 62 & 81.

Regarding claims 41, 62 & 81:

*Shafer et al.* describes, wherein the Data Partitioning function selects one or more data partitions from a table using a database internal hashing technique. **[(2.3 Performing the split, page 5, "*(hash table)*")]** It would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matters pertains, to employ a internal hashing techniques, because hashing implies mapping a numerical value by a transformation i.e., hashing is used to convert an identifier or key, typically meaning to a user, into a value for the location of the corresponding data in a structure e.g., such as a table.

[redacted]

## Claim Rejections - 35 USC § 103

13. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

14. Claims 44 & 45 are rejected under 35 U.S.C. 103(a) as being unpatentable over *Iyer et al.* (USPN 5,899,992) in view of *Bridges* (USPN 5,548,770).

Regarding claim 44:

*Iyer et al.* teaches:

A method for performing data mining applications, comprising:

**(a)** storing a relational database on one or more data storage devices connected to a computer **[(FIG. 1;** item **104** *random access memory (RAM)*) **& (col. 3, line 9-29 "*The RDBMS software 108*** *receives commands from users for performing various search and retrieval functions, termed queries, against one or more databases 112 stored in the data storage devices 106. In the preferred embodiment, these queries conform to the Structured Query Language (SQL) standard, although other types of queries could also be used without departing from the scope of the invention. The queries invoke functions performed by the* ***RDBMS software 108****, such as definition, access control, interpretation, compilation, database retrieval, and update of user and system data.* ***Generally, the RDBMS software 108****, the SQL queries, and the instructions derived therefrom, are all tangibly embodied in or readable from a computer-readable medium, e.g. one or more of the data storage devices 106 and/or data communications devices coupled to the*

*computer.* ***Moreover, the RDBMS software 108****, the SQL queries, and the instructions derived therefrom, are all comprised of instructions which, when read and executed by the computer 100, causes the computer 100 to perform the steps necessary to implement and/or use the present invention.***")]**;

**(b)** accessing the relational database stored on the data storage devices using a relational database management system **[(col. 2, line 54 to col. 3, line 9** "*FIG. 1 is a block diagram illustrating an exemplary hardware environment used to implement the preferred embodiment of the invention. In the exemplary environment, a computer 100 is comprised of one or more processors 102,* ***random access memory (RAM) 104, and assorted peripheral devices.*** *The peripheral devices usually include one or more fixed and/or removable data storage devices 106, such as a hard disk, floppy disk, CD-ROM, tape, etc. Those skilled in the art will recognize that any combination of the above components, or any number of different components, peripherals, and other devices, may be used with the computer 100. The present invention is typically implemented using* ***relational database management system (RDBMS)*** *software 108, such as the DB2 product sold by IBM Corporation, although it may be implemented with any database management system (DBMS) software. The RDBMS software 108 executes under the control of an operating system 110, such MVS, AIX, OS/2, WINDOWS NT, WINDOWS, UNIX, etc. Those skilled in the art will recognize that any combination of the software, or any number of different software, may be used to implement the present invention.***")]**; and

*Iyer et al.* does not explicitly teach: **(c)** … massively parallel relational database management system, … by the relational database management system. However, *Bridges* teaches: **(c)** … massively parallel relational database management system, … by the relational database

management system. **[(col. 4, line 29-65** "*Referring now to the drawings, FIG. 1 illustrates the database and indexing system 10 of the present invention. A conventional* ***Relational Database Management System (RDBMS) server*** *12 is provided. Illustratively, server 12 may be an Alpha AXP available from Digital Equipment Corporation. Server 12 includes a software component which is a Standard Query Language (SQL) Engine 14. SQL engine 14 runs on top of an operating system and connects low level data tables to end users and to various* ***RDBMS*** *tool kits. The physical computer 16 is a minicomputer or mainframe computer. Computer 16 and SQL engine 14 are jointly referred to as* ***RDBMS*** *server 12. Computer 16 includes a memory, a CPU, buses, and I/O capabilities. Preferably, server 12 has fast I/O channels coupled to a large disk array or disk farm 18. Elements 12-18 are components normally associated with traditional* ***RDBMS****. Data is stored on disk system 18 in a record based format with serial input and output. The present invention adds four new components to the* ***traditional RDBMS core system****.* ***A parallel computer 20 is coupled to server 12****. Illustratively,* ***parallel computer 20*** *may be a MP-1216 available from MasPar Computer Corporation. Parallel computer 20 must be a parallel processor computer to support the functionality of the present invention. Preferably,* ***computer 20 is a massively parallel processor (MPP) having more than 1000 processors. Parallel computer 20*** *contains the same components as a standard computer system, including memory, CPUs, buses and I/O capabilities.* ***Parallel computer 20 is coupled to a parallel disk array 22.*** *Advantageously, parallel computer 20 can take advantage of an increased I/O bandwidth associated with parallel computers and parallel disk arrays. The relative size of parallel disk array 22 is smaller than the conventional disk system 18 since, in the present invention, only indexes must be stored in the parallel disk array 22. It is understood, however, that all the data and not just indexes may*

*be stored in parallel disk array 22, if desired.*")] It would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matters pertains, to employ a massively parallel relational database management system for the purpose handling large amounts of data residing on a relational database management system e.g., a relational database management system, a relational database management system sold under the ORACLE7™ which provides simple operations which perform the mass transfer of database information from one database to another. Combined, with a multiprocessor systems i.e., a massively parallel processing systems comprising a plurality of individual processors, each having its own CPU and memory, organized in a loosely coupled environment, or a distributed processing system operating in a loosely coupled environment, for example, over a local area network. The ability to process massive amounts of information in a rapid and efficient mode of processing is quickly realized.

Regarding claim 45:

*Iyer et al.* teaches:

An article of manufacture comprising logic embodying a method for performing data mining applications, comprising:

**(a)** storing a relational database on one or more data storage devices connected to a computer **[(FIG. 1;** item **104** *random access memory (RAM)*) **& (col. 3, line 9-29** ***"The RDBMS software 108*** *receives commands from users for performing various search and retrieval functions, termed queries, against one or more databases 112 stored in the data storage devices 106. In the preferred embodiment, these queries conform to the Structured Query Language (SQL) standard,*

*although other types of queries could also be used without departing from the scope of the invention. The queries invoke functions performed by the* ***RDBMS software 108****, such as definition, access control, interpretation, compilation, database retrieval, and update of user and system data.* ***Generally, the RDBMS software 108****, the SQL queries, and the instructions derived therefrom, are all tangibly embodied in or readable from a computer-readable medium, e.g. one or more of the data storage devices 106 and/or data communications devices coupled to the computer.* ***Moreover, the RDBMS software 108****, the SQL queries, and the instructions derived therefrom, are all comprised of instructions which, when read and executed by the computer 100, causes the computer 100 to perform the steps necessary to implement and/or use the present invention.*"**)];**

**(b)** accessing the relational database stored on the data storage devices using a relational database management system **[(col. 2, line 54 to col. 3, line 9** "*FIG. 1 is a block diagram illustrating exemplary hardware environment used to implement the preferred embodiment of the invention. In the exemplary environment, a computer 100 is comprised of one or more processors 102,* ***random access memory (RAM) 104, and assorted peripheral devices.*** *The peripheral devices usually include one or more fixed and/or removable data storage devices 106, such as a hard disk, floppy disk, CD-ROM, tape, etc. Those skilled in the art will recognize that any combination of the above components, or any number of different components, peripherals, and other devices, may be used with the computer 100. The present invention is typically implemented using* ***relational database management system (RDBMS)*** *software 108, such as the DB2 product sold by IBM Corporation, although it may be implemented with any database management system (DBMS) software. The RDBMS software 108 executes under the control of an operating*

*system 110, such MVS, AIX, OS/2, WINDOWS NT, WINDOWS, UNIX, etc. Those skilled in the art will recognize that any combination of the software, or any number of different software, may be used to implement the present invention.*“)]; and

*Iyer et al.* does not explicitly teach: **(c)** … massively parallel relational database management system, … by the relational database management system. However, *Bridges* teaches: **(c)** … massively parallel relational database management system, … by the relational database management system. **[(col. 4, line 29-65** “*Referring now to the drawings, FIG. 1 illustrates the database and indexing system 10 of the present invention. A conventional* ***Relational Database Management System (RDBMS) server*** *12 is provided. Illustratively, server 12 may be an Alpha AXP available from Digital Equipment Corporation. Server 12 includes a software component which is a Standard Query Language (SQL) Engine 14. SQL engine 14 runs on top of an operating system and connects low level data tables to end users and to various* ***RDBMS*** *tool kits. The physical computer 16 is a minicomputer or mainframe computer. Computer 16 and SQL engine 14 are jointly referred to as* ***RDBMS*** *server 12. Computer 16 includes a memory, a CPU, buses, and I/O capabilities. Preferably, server 12 has fast I/O channels coupled to a large disk array or disk farm 18. Elements 12-18 are components normally associated with traditional* ***RDBMS****. Data is stored on disk system 18 in a record based format with serial input and output. The present invention adds four new components to the* ***traditional RDBMS core system. A parallel computer 20 is coupled to server 12****. Illustratively,* ***parallel computer 20*** *may be a MP-1216 available from MasPar Computer Corporation. Parallel computer 20 must be a parallel processor computer to support the functionality of the present invention. Preferably,* ***computer 20 is a massively parallel processor (MPP) having more than 1000 processors. Parallel computer 20***

*contains the same components as a standard computer system, including memory, CPUs, buses and I/O capabilities.* ***Parallel computer 20 is coupled to a parallel disk array 22.*** *Advantageously, parallel computer 20 can take advantage of an increased I/O bandwidth associated with parallel computers and parallel disk arrays. The relative size of parallel disk array 22 is smaller than the conventional disk system 18 since, in the present invention, only indexes must be stored in the parallel disk array 22. It is understood, however, that all the data and not just indexes may be stored in parallel disk array 22, if desired.*")] It would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matters pertains, to employ a massively parallel relational database management system for the purpose handling large amounts of data residing on a relational database management system e.g., a relational database management system, a relational database management system sold under the ORACLE7™ which provides simple operations which perform the mass transfer of database information from one database to another. Combined, with a multiprocessor systems i.e., a massively parallel processing systems comprising a plurality of individual processors, each having its own CPU and memory, organized in a loosely coupled environment, or a distributed processing system operating in a loosely coupled environment, for example, over a local area network. The ability to process massive amounts of information in a rapid and efficient mode of processing is quickly realized.

## Correspondence Information

15. Any inquires concerning this communication or earlier communications from the examiner should be directed to Michael B. Holmes, who may be reached Monday through Friday, between 8:00 a.m. and 5:00 p.m. EST. or via telephone at (571) 272-3686 or facsimile transmission (571) 273-3686 or email Michael.holmesb@uspto.gov.

If you need to send an Official facsimile transmission, please send it to (703) 746-7239.

If attempts to reach the examiner are unsuccessful the Examiner's Supervisor, Anthony Knight, may be reached at (571) 272-3687.

Hand-delivered responses should be delivered to the Receptionist @ (Customer Service Window Randolph Building 401 Dulany Street Alexandria, VA 22313), located on the first floor of the south side of the Randolph Building.

***Michael B. Holmes***
Patent Examiner
Artificial Intelligence
Art Unit 2121
United States Department of Commerce
Patent & Trademark Office

*Sunday, April 03, 2005*

*MBH*